

CLAIM LISTING

1. (Currently Amended) A method for translating ~~Verilog~~ a hardware structure design language into ~~C++~~ a general purpose programming language, the method comprising:

searching for a ~~Verilog~~ a hardware structure design language pattern in a ~~Verilog~~ a hardware structure design language file, the ~~Verilog~~ hardware structure design language pattern associated with a specific functionality; and

substituting the ~~Verilog~~ a hardware structure design language pattern with a ~~C++~~ a general purpose programming language expression, wherein the ~~C++~~ a general purpose programming language expression is associated with the specific functionality;

if the hardware structure design language file comprises a task library:

identifying a hardware structure design language task within the task library; and

translating the hardware structure design language task into a general purpose programming language function; and

if the hardware structure design language file comprises a main driver:

inserting in the hardware structure design language file at least one general purpose programming language interface header.

2. (Currently Amended) The method of claim 1, wherein the translating from ~~Verilog~~ the hardware structure design language into ~~C~~ the general purpose programming language utilizes macro functions in VBScript.

3. (Currently Amended) The method of claim 2, further comprising identifying whether the ~~Verilog hardware structure design language~~ file comprises at least one of a task library, a main driver, and a driver module.

4. (Cancelled)

5. (Currently Amended) The method of claim 1, wherein the ~~Verilog hardware structure design language~~ pattern comprises # delay statements from the ~~Verilog hardware structure design language~~ file.

6. (Currently Amended) The method of claim 1, wherein the ~~Verilog hardware structure design language~~ pattern comprises `ifdef statements in the ~~Verilog hardware structure design language~~ file.

7. (Currently Amended) The method of claim 1, wherein the ~~Verilog hardware structure design language~~ pattern comprises ` symbols from the ~~Verilog hardware structure design language~~ file.

8. (Currently Amended) The method of claim 1, wherein the ~~Verilog~~ hardware structure design language pattern comprises a begin keyword in the ~~Verilog~~ hardware structure design language file to a "{" symbol.

9. (Currently Amended) The method of claim 1, wherein the ~~Verilog~~ hardware structure design language pattern comprises an end keyword in the ~~Verilog~~ hardware structure design language file to a "}" symbol.

10. (Currently Amended) The method of claim 1, wherein the ~~Verilog~~ hardware structure design language pattern comprises at least one register definition in the ~~Verilog~~ hardware structure design language file into at least one ~~C++~~ general purpose programming language definition.

11. (Currently Amended) The method of claim 1, wherein the ~~Verilog~~ hardware structure design language pattern comprises at least one combinatorial assignment in the ~~Verilog~~ hardware structure design language file.

12. (Currently Amended) The method of claim 1, wherein the ~~Verilog~~ hardware structure design language pattern comprises at least one event in the ~~Verilog~~ hardware structure design language file into at least one ~~C++~~ general purpose programming language event.

13. (Currently Amended) The method of claim 1, wherein the ~~Verilog hardware structure design language~~ pattern comprises at least one ~~Verilog hardware structure design language~~ switch in the ~~Verilog hardware structure design language~~ file into at least one ~~C++ general purpose programming language~~ switch.

14. (Currently Amended) The method of claim 1, wherein the ~~Verilog hardware structure design language~~ pattern comprises at least one ~~Verilog hardware structure design language~~ concat expressions in the ~~Verilog hardware structure design language~~ file into at least one ~~C++ general purpose programming language~~ concat expressions.

15. (Currently Amended) The method of claim 1, wherein the ~~Verilog hardware structure design language~~ pattern comprises at least one ~~Verilog hardware structure design language~~ parameter in the ~~Verilog hardware structure design language~~ file into at least one ~~C++ general purpose programming language~~ #define.

16. (Currently Amended) The method of claim 1, wherein the ~~Verilog hardware structure design language~~ pattern comprises at least one ~~Verilog hardware structure design language~~ const in the ~~Verilog hardware structure design language~~ file into at least one ~~C++ general purpose programming language~~ const.

17. (Currently Amended) The method of claim 1, wherein the ~~Verilog hardware structure design language~~ pattern comprises at least one ~~Verilog hardware structure design language~~ bit access macro in the ~~Verilog hardware structure design language~~ file into at least one ~~C++~~ general purpose programming language functional equivalent.

18. (Currently Amended) A machine-readable storage having stored thereon, a computer program having at least one code section for translating ~~Verilog a hardware structure design language~~ to ~~C++ a general purpose programming language~~, at least one code section being executable by a machine for causing the machine to perform steps comprising:

searching for ~~Verilog a hardware structure design language~~ pattern in ~~Verilog a hardware structure design language~~ file, the ~~Verilog hardware structure design language~~ pattern associated with a specific functionality; and

substituting the ~~Verilog hardware structure design language~~ pattern with a ~~C++ general purpose programming language~~ expression, wherein the ~~C++ general purpose programming language~~ expression is associated with the specific functionality; and

if the hardware structure design language file comprises a task library:

code for identifying a hardware structure design language task within the task library; and

code for translating the hardware structure design language task into the general purpose programming language function; and

if the hardware structure design language file comprises a main driver:

code for inserting in the hardware structure design language file at least one general purpose programming language interface header.

19. (Currently Amended) The machine-readable storage according to claim 18, wherein the translating from ~~Verilog~~ the hardware structure design language into ~~C++~~ the general purpose programming language utilizes macro functions in VBScript.

20. (Currently Amended) The machine-readable storage according to claim 19, further comprising code for identifying whether the ~~Verilog~~ hardware structure design language file comprises at least one of a task library, a main driver, and a driver module.

21. (Cancelled).

22. (Currently Amended) The machine-readable storage according to claim 20, further comprising code for removing # delay statements from the ~~Verilog~~ hardware structure design language file.

23. (Currently Amended) The machine-readable storage according to claim 20, further comprising code for translating `ifdef statements in the ~~Verilog hardware structure design language~~ file.

24. (Currently Amended) The machine-readable storage according to claim 20, further comprising code for removing ` symbols from the ~~Verilog hardware structure design language~~ file.

25. (Currently Amended) The machine-readable storage according to claim 20, further comprising code for converting a begin keyword in the ~~Verilog hardware structure design language~~ file to a "{" symbol.

26. (Currently Amended) The machine-readable storage according to claim 20, further comprising code for converting an end keyword in the ~~Verilog hardware structure design language~~ file to a "}" symbol.

27. (Currently Amended) The machine-readable storage according to claim 20, further comprising code for converting at least one register definition in the ~~Verilog hardware structure design language~~ file into at least one ~~C++ general purpose programming language~~ definition.

28. (Currently Amended) The machine-readable storage according to claim 20, further comprising code for

performing at least one combinatorial assignment in the ~~Verilog-hardware structure design language~~ file.

29. (Currently Amended) The machine-readable storage according to claim 20, further comprising code for converting at least one event in the ~~Verilog-hardware structure design language~~ file into at least one ~~C++ general purpose programming language~~ event.

30. (Currently Amended) The machine-readable storage according to claim 20, further comprising code for converting at least one ~~Verilog-hardware structure design language~~ switch in the ~~Verilog-hardware structure design language~~ file into at least one ~~C++ general purpose programming language~~ switch.

31. (Currently Amended) The machine-readable storage according to claim 20, further comprising code for converting at least one ~~Verilog-hardware structure design language~~ concat expressions in the ~~Verilog-hardware structure design language~~ file into at least one ~~C++ general purpose programming language~~ concat expressions.

32. (Currently Amended) The machine-readable storage according to claim 20, further comprising code for converting at least one ~~Verilog-hardware structure design language~~ parameter in the ~~Verilog-hardware structure design language~~ file into at least one ~~C++ general purpose programming language~~ #define.

33. (Currently Amended) The machine-readable storage according to claim 20, further comprising code for converting at least one Verilog hardware structure design language const in the Verilog hardware structure design language file into at least one C++ general purpose programming language const.

34. (Currently Amended) The machine-readable storage according to claim 20, further comprising code for converting at least one Verilog hardware structure design language bit access macro in the Verilog hardware structure design language file into at least one C++ general purpose programming language functional equivalent.